Базовые понятия языков программирования

Типы данных в Python

```
a = 10          # int
b = 3.14        # float
c = "Hi"        # str
d = True        # bool
e = {k:v, k2:v}     # dict
f = [1, 2, 3]       # list   изменяемый,   неуникальный, упорядоченный
g = {1, 2, 3}       # set    изменяемый,   уникальный,   сортируемый
h = (1, "hi", True)# tuple  неизменяемый, неуникальный, упорядоченный
```

Переменные как объект в языке Python

- В Python переменные являются ссылками на объекты в памяти. Это значит, что переменная не “хранит” значение, а ссылается на объект с этим значением.

```
x = 5
y = x # Теперь y ссылается на тот же объект, что и x
```

Из-за этого изменения одного объекта могут повлиять на другие переменные, если они ссылаются на один и тот же изменяемый объект (например, список).

```
a = [1, 2, 3]
b = a
b.append(4)
print(a) # Выведет [1, 2, 3, 4], так как b и a указывают на один список.
```

Имена переменных и зарезервированные слова

- Имена переменных в python должны:
  - начинаться с буквы или _, но не с цифры,
  - содержать только буквы, цифры и _,
  - не совпадать с зарезервированными словами (**`if`**, **`else`**, **`while`**, **`def`** и др.).

- Пример корректных имен:

```
my_var = 10
_user = "Alice"
value123 = 3.14
```

- Пример некорректных имен:

```
3var = 5   # Ошибка: имя не может начинаться с цифры
if = 10    # Ошибка: имя совпадает с ключевым словом
```

Инструкции

- В Python инструкции — это команды, которые выполняются интерпретатором. Они могут быть простыми (одна строка) или сложными (с блоками кода).
- Пример простой инструкции:
```
print("Hello, world!")  # Выводит строку на экран
```
- Пример сложной инструкции с блоком кода:
```
if 5 > 3:
    print("Yes") # Вложенный блок кода
```

Операторы и операнды

- Операторы — это символы, которые выполняют операции над операндами (значениями).

- Примеры операторов:
  Арифметические: **+**, **-**, *****, **/**, **//**, **%**, ******.
  Сравнения: **==**, **!=**, **<**, **>**, **<=**, **>=**.
  Логические: **and**, **or**, **not**.
  Присваивания: **=**, **+=**, **-=**, ***=**, **/=**, **//=**, **%=**, ****=**.

- Пример:

```
x = 10 + 5 # оператор "+" складывает 10 и 5
print(x > 10) # оператор ">" сравнивает x и 10
```

Приоритеты операторов

- Операторы выполняются в порядке приоритета, например:
  1) **`**`** (возведение в степень)
  2) **`*`**, **`/`**, **`//`**, **`%`** (умножение, деление)
  3) **`+`**, **`-`** (сложение, вычитание)

- Пример:
  `print(2 + 3 * 4)` `# Сначала умножение (3*4), затем сложение (2+12), результат 14.`

- Чтобы изменить приоритет, используют скобки:
  `print((2 + 3) * 4)` `# Теперь сначала сложение (2+3), затем умножение, результат 20.`

Операции над переменными

- Переменные можно изменять с помощью операторов:

```
x = 5
x += 3  # эквивалентно x = x + 3
print(x)  # Выведет 8
```

Порядок выполнения программы

- Программа выполняется сверху вниз, если нет циклов, условий или функций. Например:

```
print("Начало")
print("Конец")
```

- Но при наличии условных операторов и циклов порядок выполнения может изменяться.

```
x = 10
if x > 5:
    print("Больше 5")  # Этот код выполнится, если x больше 5
```

Ввод/вывод

- В Python ввод данных осуществляется с помощью input(), а вывод — через print().

```
name = input("Введите ваше имя: ")
print("Привет,", name)
```

Преобразование типов

- Чтобы изменить тип данных, используют приведение типов:

```
x = "10"
y = int(x)   # Преобразование строки "10" в число 10
a = int("5")       # 5 (из строки в число)
b = str(123)       # "123" (из числа в строку)
c = float("3.14") # 3.14 (из строки в число с плавающей точкой)
```

Ошибки: синтаксические и логические

- Синтаксические ошибки – возникают, когда код написан с нарушением правил Python. Например:

```
if 5 > 3  # Ошибка: пропущен двоеточие
    print(“Yes")
```

- Логические ошибки – код выполняется без ошибок, но результат неверный. Например:

```
x = 10
y = 5
print("Сумма:", x - y)  # Ошибка: должно быть x + y
```